# Assignment 8: Time Series Analysis

Aileen Lavelle

Spring 2023

## OVERVIEW

This exercise accompanies the lessons in Environmental Data Analytics on generalized linear models.

## Directions

1. Rename this file `<FirstLast>_A08_TimeSeries.Rmd` (replacing `<FirstLast>` with your first and last name).
2. Change "Student Name" on line 3 (above) with your name.
3. Work through the steps, **creating code and output** that fulfill each instruction.
4. Be sure to **answer the questions** in this assignment document.
5. When you have completed the assignment, **Knit** the text and code into a single PDF file.

## Set up

1. Set up your session:

- Check your working directory
- Load the tidyverse, lubridate, zoo, and trend packages
- Set your ggplot theme

```
#1
library(tidyverse);library(lubridate);library(here);library(trend);library(Kendall)
```

```
## -- Attaching packages --------------------------------------- tidyverse 1.3.2 --
## v ggplot2 3.4.0     v purrr   1.0.1
## v tibble  3.1.8     v dplyr   1.1.0
## v tidyr   1.3.0     v stringr 1.5.0
## v readr   2.1.3     v forcats 1.0.0
## -- Conflicts ------------------------------------------ tidyverse_conflicts() --
## x dplyr::filter() masks stats::filter()
## x dplyr::lag()    masks stats::lag()
##
## Attaching package: 'lubridate'
##
##
## The following objects are masked from 'package:base':
##
##     date, intersect, setdiff, union
```

```
##
##
## here() starts at /Users/aileen/Desktop/Duke/Environmental_Data_Analytics/EDA_Spring_2023_corrected/E
```

```
here()
```

```
## [1] "/Users/aileen/Desktop/Duke/Environmental_Data_Analytics/EDA_Spring_2023_corrected/EDA-Spring202
```

```
Aileentheme <- theme_classic(base_size = 14) +
  theme(axis.text = element_text(color = "black"),
        axis.ticks = element_line(color = "black"),
        plot.background = element_rect(color= "white"))
```

2. Import the ten datasets from the Ozone_TimeSeries folder in the Raw data folder. These contain ozone concentrations at Garinger High School in North Carolina from 2010-2019 (the EPA air database only allows downloads for one year at a time). Import these either individually or in bulk and then combine them into a single dataframe named `GaringerOzone` of 3589 observation and 20 variables.

```
#2
epadata2010 <- read.csv(here("Data","Raw","Ozone_TimeSeries","EPAair_O3_GaringerNC2010_raw.csv"))
epadata2011 <- read.csv(here("Data","Raw","Ozone_TimeSeries","EPAair_O3_GaringerNC2011_raw.csv"))
epadata2012 <- read.csv(here("Data","Raw","Ozone_TimeSeries","EPAair_O3_GaringerNC2012_raw.csv"))
epadata2013 <- read.csv(here("Data","Raw","Ozone_TimeSeries","EPAair_O3_GaringerNC2013_raw.csv"))
epadata2014 <- read.csv(here("Data","Raw","Ozone_TimeSeries","EPAair_O3_GaringerNC2014_raw.csv"))
epadata2015 <- read.csv(here("Data","Raw","Ozone_TimeSeries","EPAair_O3_GaringerNC2015_raw.csv"))
epadata2016 <- read.csv(here("Data","Raw","Ozone_TimeSeries","EPAair_O3_GaringerNC2016_raw.csv"))
epadata2017 <- read.csv(here("Data","Raw","Ozone_TimeSeries","EPAair_O3_GaringerNC2017_raw.csv"))
epadata2018 <- read.csv(here("Data","Raw","Ozone_TimeSeries","EPAair_O3_GaringerNC2018_raw.csv"))
epadata2019 <- read.csv(here("Data","Raw","Ozone_TimeSeries","EPAair_O3_GaringerNC2019_raw.csv"))
```

## Wrangle

3. Set your date column as a date class.

4. Wrangle your dataset so that it only contains the columns Date, Daily.Max.8.hour.Ozone.Concentration, and DAILY_AQI_VALUE.

5. Notice there are a few days in each year that are missing ozone concentrations. We want to generate a daily dataset, so we will need to fill in any missing days with NA. Create a new data frame that contains a sequence of dates from 2010-01-01 to 2019-12-31 (hint: `as.data.frame(seq())`). Call this new data frame Days. Rename the column name in Days to "Date".

6. Use a `left_join` to combine the data frames. Specify the correct order of data frames within this function so that the final dimensions are 3652 rows and 3 columns. Call your combined data frame GaringerOzone.

```
#3
#correct dates
epadata2010  <- epadata2010 %>%
  mutate(Date = mdy(Date))
epadata2011  <- epadata2011 %>%
  mutate(Date = mdy(Date))
```

```
epadata2012  <- epadata2012 %>%
  mutate(Date = mdy(Date))
epadata2013  <- epadata2013 %>%
  mutate(Date = mdy(Date))
epadata2014  <- epadata2014 %>%
  mutate(Date = mdy(Date))
epadata2015  <- epadata2015 %>%
  mutate(Date = mdy(Date))
epadata2016  <- epadata2016%>%
  mutate(Date = mdy(Date))
epadata2017  <- epadata2017 %>%
  mutate(Date = mdy(Date))
epadata2018  <- epadata2018 %>%
  mutate(Date = mdy(Date))
epadata2019  <- epadata2019 %>%
  mutate(Date = mdy(Date))
#4
#select columns for all data
epadata2010  <- epadata2010 %>%
  select(Date,Daily.Max.8.hour.Ozone.Concentration, DAILY_AQI_VALUE)
epadata2011  <- epadata2011 %>%
  select(Date,Daily.Max.8.hour.Ozone.Concentration, DAILY_AQI_VALUE)
epadata2012  <- epadata2012 %>%
  select(Date,Daily.Max.8.hour.Ozone.Concentration, DAILY_AQI_VALUE)
epadata2013  <- epadata2013 %>%
  select(Date,Daily.Max.8.hour.Ozone.Concentration, DAILY_AQI_VALUE)
epadata2014  <- epadata2014 %>%
  select(Date,Daily.Max.8.hour.Ozone.Concentration, DAILY_AQI_VALUE)
epadata2015  <- epadata2015 %>%
  select(Date,Daily.Max.8.hour.Ozone.Concentration, DAILY_AQI_VALUE)
epadata2016  <- epadata2016 %>%
  select(Date,Daily.Max.8.hour.Ozone.Concentration, DAILY_AQI_VALUE)
epadata2017  <- epadata2017 %>%
  select(Date,Daily.Max.8.hour.Ozone.Concentration, DAILY_AQI_VALUE)
epadata2018  <- epadata2018 %>%
  select(Date,Daily.Max.8.hour.Ozone.Concentration, DAILY_AQI_VALUE)
epadata2019  <- epadata2019 %>%
  select(Date,Daily.Max.8.hour.Ozone.Concentration, DAILY_AQI_VALUE)
#5
#create Day dataframe
Days <-  as.data.frame(rep(seq(as.Date('2010-01-01'), as.Date('2019-12-31'), by = 'days')))
#rename column to date
colnames(Days)[1] <- 'Date'

#6
Garingercombined <- bind_rows(epadata2010, epadata2011, epadata2012, epadata2013, epadata2014, epadata20
GaringerOzone <- left_join(Days, Garingercombined, by='Date')
```

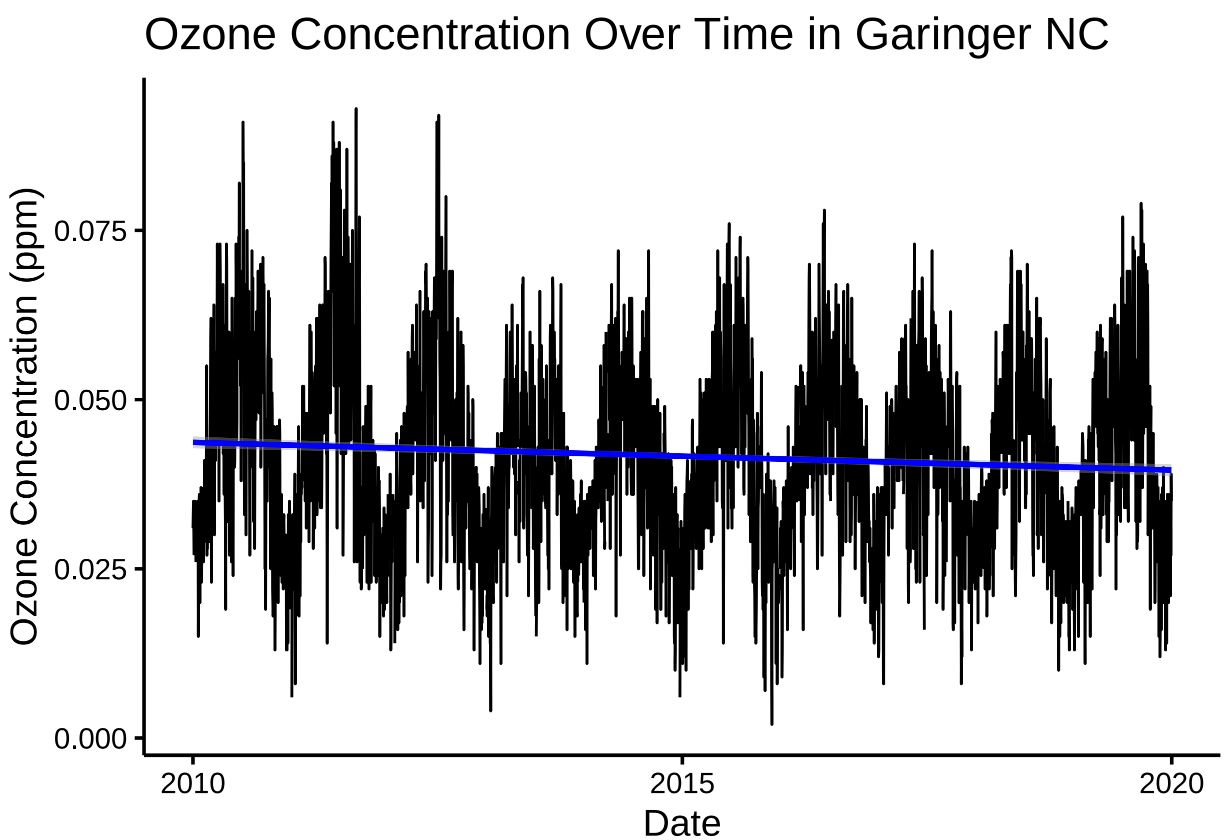
## Visualize

7. Create a line plot depicting ozone concentrations over time. In this case, we will plot actual concentrations in ppm, not AQI values. Format your axes accordingly. Add a smoothed line showing any linear

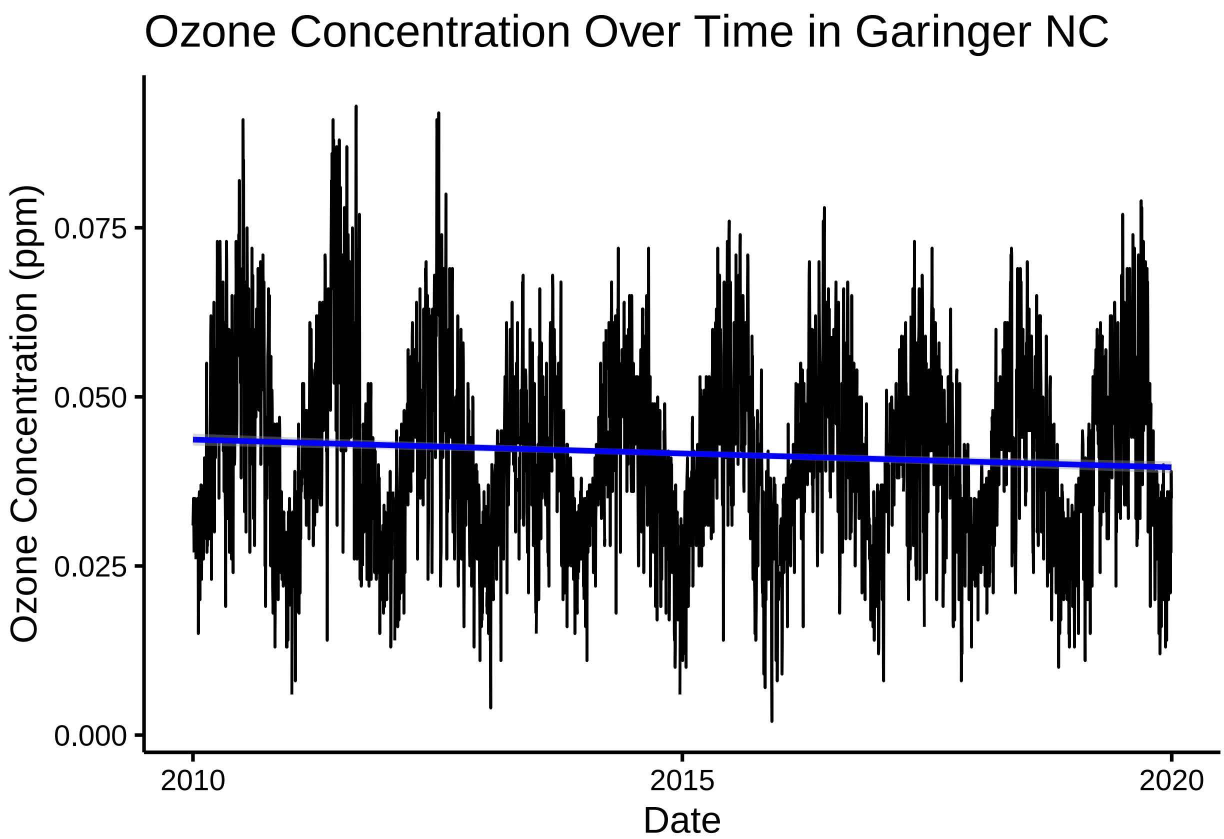
trend of your data. Does your plot suggest a trend in ozone concentration over time?

```
#7
Ozoneovertime <-
  ggplot(GaringerOzone, aes(x=Date, y=Daily.Max.8.hour.Ozone.Concentration)) +
  geom_line()+
  geom_smooth(formula =y ~ x, color= "blue") +
  labs(
    title = "Ozone Concentration Over Time in Garinger NC",
    x= "Date",
    y= "Ozone Concentration (ppm)") +
  Aileentheme
plot(Ozoneovertime) + Aileentheme
```

```
## `geom_smooth()` using method = 'gam'

## Warning: Removed 63 rows containing non-finite values (`stat_smooth()`).
```




```
## `geom_smooth()` using method = 'gam'

## Warning: Removed 63 rows containing non-finite values (`stat_smooth()`).
```




Answer: This plot depicts a slight downward trend of ozone concentration over time. The

### Time Series Analysis

Study question: Have ozone concentrations changed over the 2010s at this station?

8. Use a linear interpolation to fill in missing daily data for ozone concentration. Why didn't we use a piecewise constant or spline interpolation?

```
#8
Garinger_Ozone_clean <-
  GaringerOzone %>%
  mutate(Daily.Max.8.hour.Ozone.Concentration.clean = zoo::na.approx(Daily.Max.8.hour.Ozone.Concentrati
  select(Date, Daily.Max.8.hour.Ozone.Concentration.clean, DAILY_AQI_VALUE)
```

Answer: We didn't use a piecewuse contstant because any missing data are assumed to be equal to the measurement made nearest to that date but we do not want to assume they are the same. We also didn't use a spline because our Ozone concentrations were linear in nature and applying a spline interpolation would use a quadratic function instead.

9. Create a new data frame called `GaringerOzone.monthly` that contains aggregated data: mean ozone concentrations for each month. In your pipe, you will need to first add columns for year and month to form the groupings. In a separate line of code, create a new Date column with each month-year combination being set as the first day of the month (this is for graphing purposes only)

```r
#9
GaringerOzoneclean <- Garinger_Ozone_clean %>%
    mutate(Month=month(Date),
         Year=year(Date)) %>%
    group_by(Month, Year) %>%
    summarise(meanOzone=mean(Daily.Max.8.hour.Ozone.Concentration.clean))
```

```
## `summarise()` has grouped output by 'Month'. You can override using the
## `.groups` argument.
```

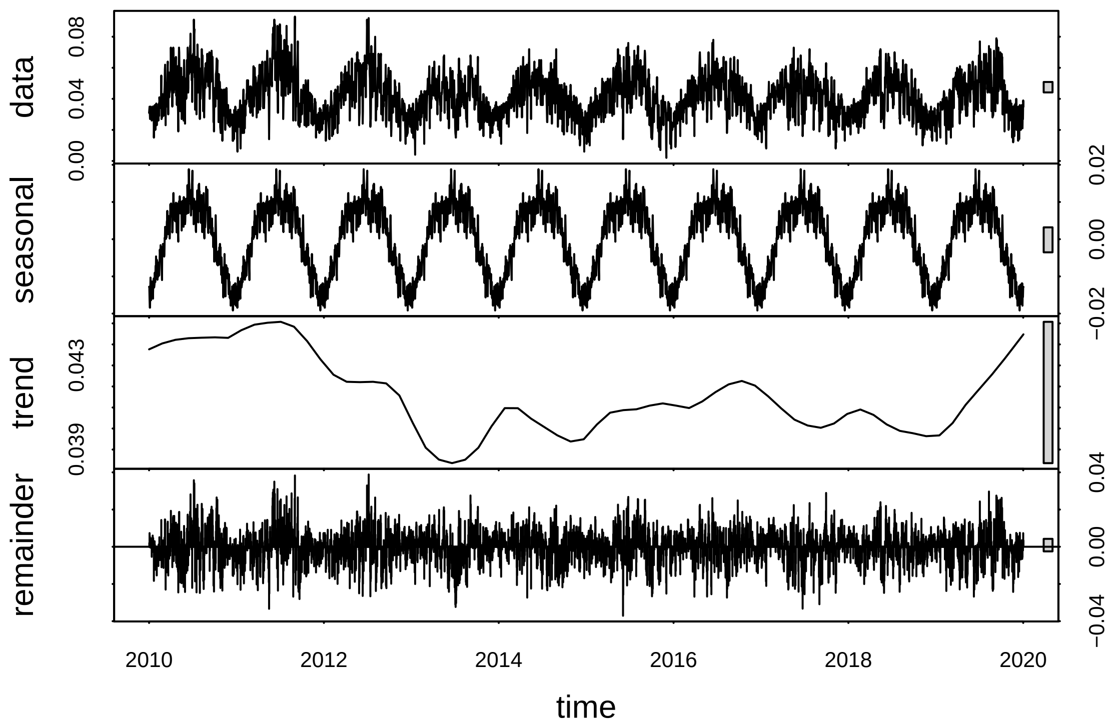
```r
GaringerOzone.monthly <- GaringerOzoneclean %>%
  mutate(Date= paste0(Month, "-",01,"-", Year))

GaringerOzone.monthly$Date <- mdy(GaringerOzone.monthly$Date)
```

10. Generate two time series objects. Name the first `GaringerOzone.daily.ts` and base it on the dataframe of daily observations. Name the second `GaringerOzone.monthly.ts` and base it on the monthly average ozone values. Be sure that each specifies the correct start and end dates and the frequency of the time series.

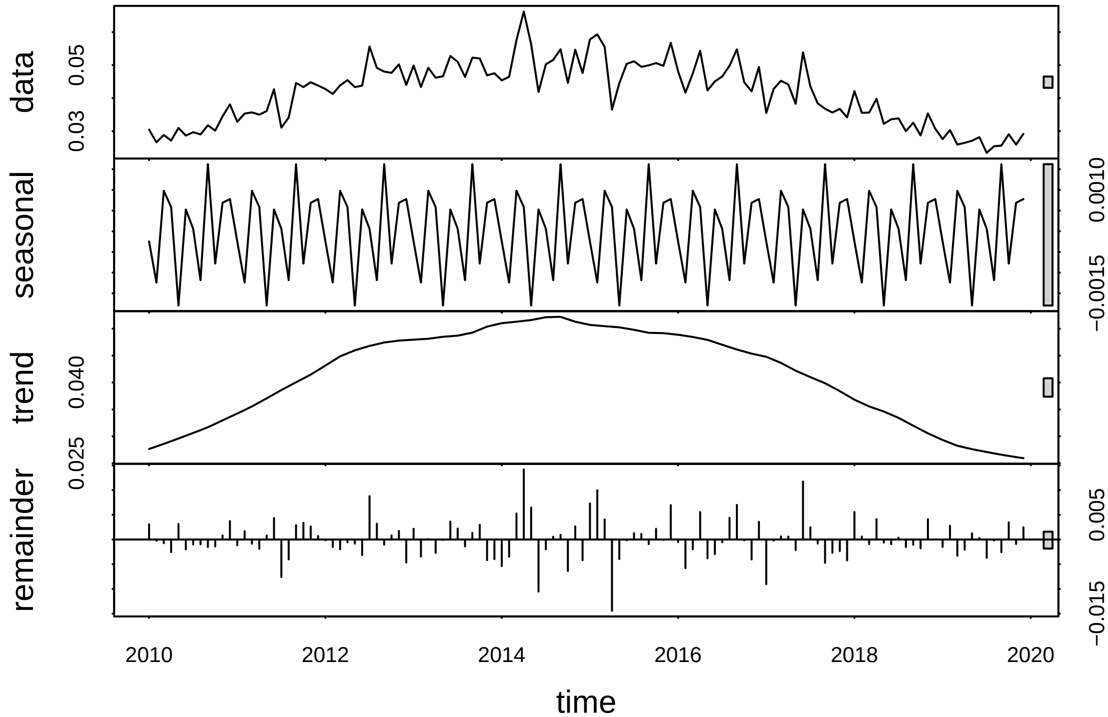
```r
#10
f_month <- month(first(GaringerOzone.monthly$Date))
f_year <- year(first(GaringerOzone.monthly$Date))

GaringerOzone.daily.ts <- ts(Garinger_Ozone_clean$Daily.Max.8.hour.Ozone.Concentration.clean,
                   start=c(f_year,f_month),
                   frequency=365)

GaringerOzone.monthly.ts <- ts(GaringerOzone.monthly$meanOzone,
                   start=c(f_year,f_month),
                   frequency=12)
```

11. Decompose the daily and the monthly time series objects and plot the components using the `plot()` function.

```r
#11
GaringerOzone.daily.decomp <- stl(GaringerOzone.daily.ts, s.window = "periodic")
plot(GaringerOzone.daily.decomp)
```




```r
GaringerOzone.monthly.decomp <- stl(GaringerOzone.monthly.ts, s.window = "periodic")
plot(GaringerOzone.monthly.decomp)
```




12. Run a monotonic trend analysis for the monthly Ozone series. In this case the seasonal Mann-Kendall is most appropriate; why is this?

```
#12
GaringerOzone.monthlytrend <- Kendall:: SeasonalMannKendall(GaringerOzone.monthly.ts)
GaringerOzone.monthlytrend
```

```
## tau = -0.1, 2-sided pvalue =0.16323
```

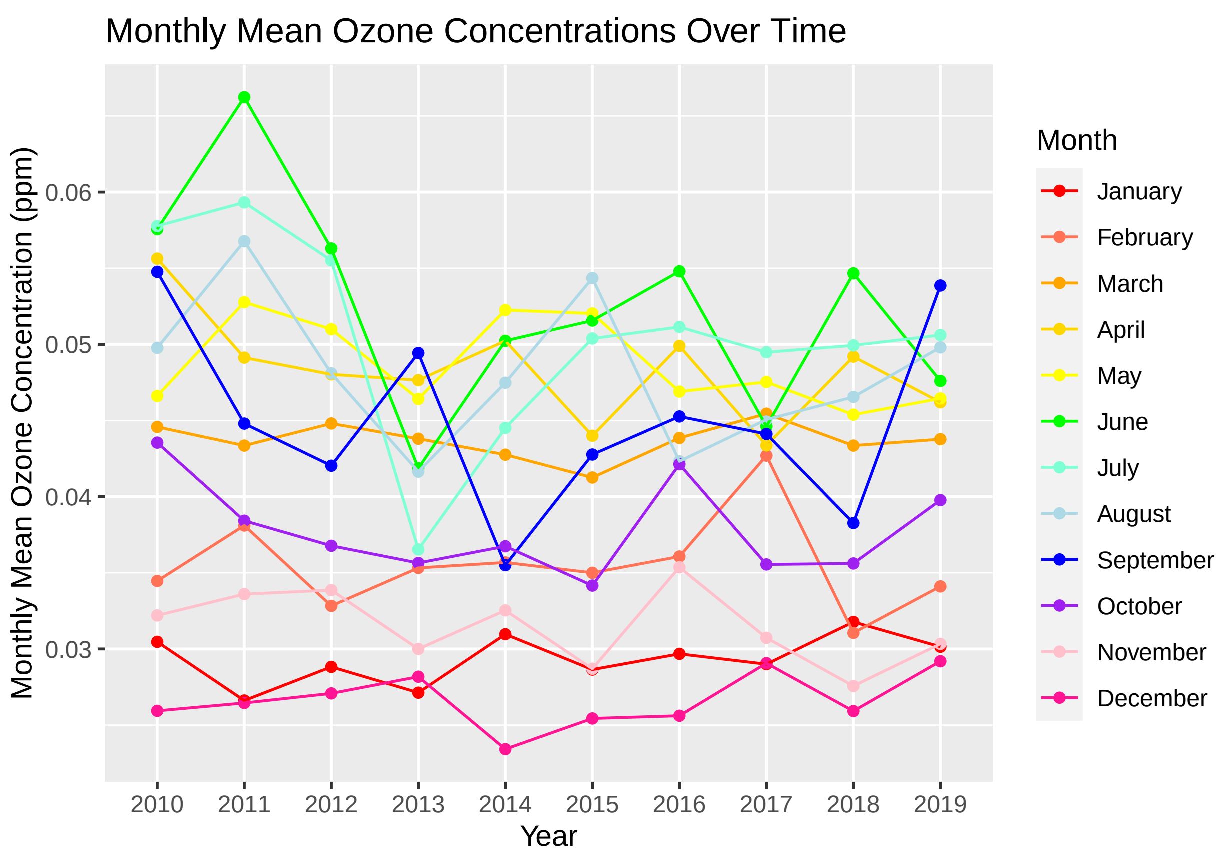
```
summary(GaringerOzone.monthlytrend)
```

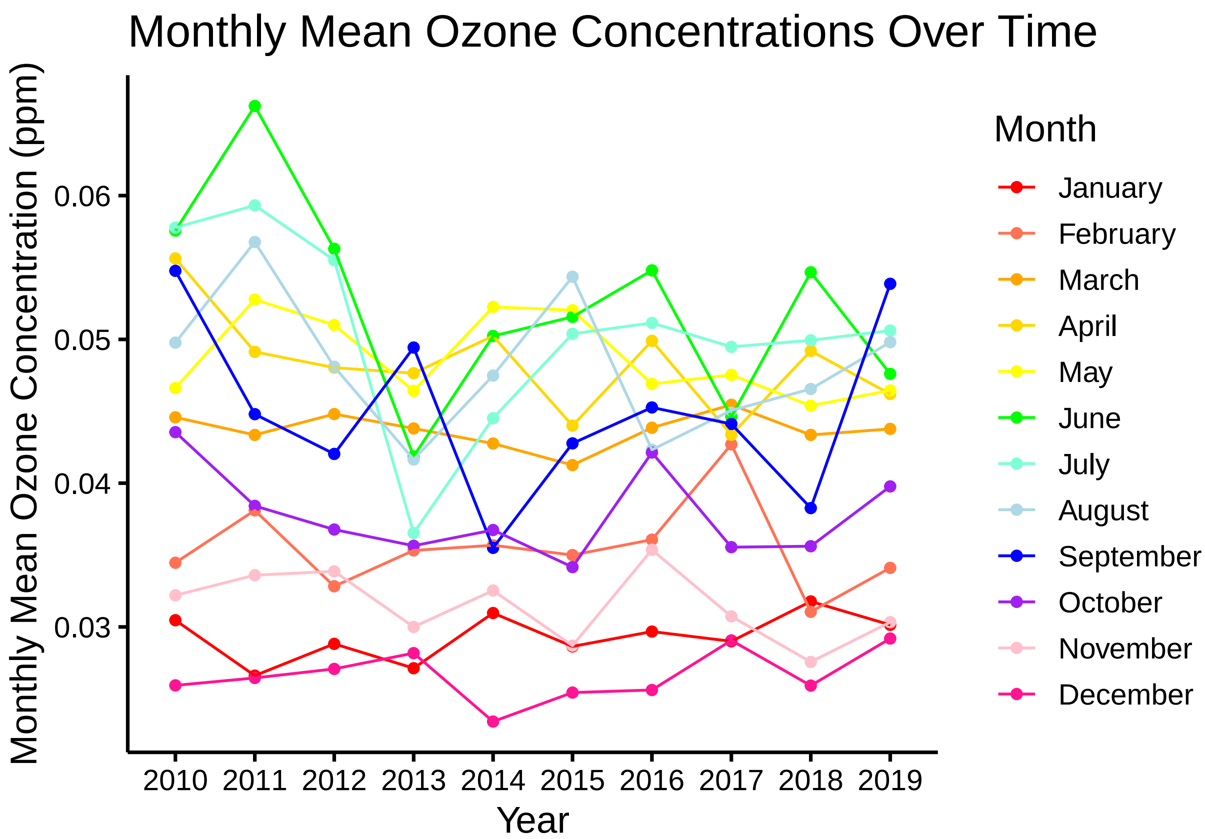
```
## Score =  -54 , Var(Score) = 1500
## denominator =  540
## tau = -0.1, 2-sided pvalue =0.16323
```

Answer: Seasonal Mann-Kendall is used in this case because it assumes seasonality, allows for missing data and is non-parametric.

13. Create a plot depicting mean monthly ozone concentrations over time, with both a geom_point and a geom_line layer. Edit your axis labels accordingly.

```
#13
#change month numbers to names
GaringerOzone.monthly$Month <-month((GaringerOzone.monthly$Month), label = TRUE, abbr = FALSE)
#create graph
MonthlyMeanOzone <-
ggplot(GaringerOzone.monthly, aes(x=factor(Year), y = meanOzone, color=as.factor(Month))) +
  geom_point() +
  geom_line(aes(group = Month)) +
  labs(
    title = "Monthly Mean Ozone Concentrations Over Time",
    x= "Year",
    y= "Monthly Mean Ozone Concentration (ppm)",
    color= "Month") +
  scale_color_manual(values = c("red", "#FF7256", "orange","#FFD700", "yellow", "green", "aquamarine",
print(MonthlyMeanOzone) + Aileentheme
```



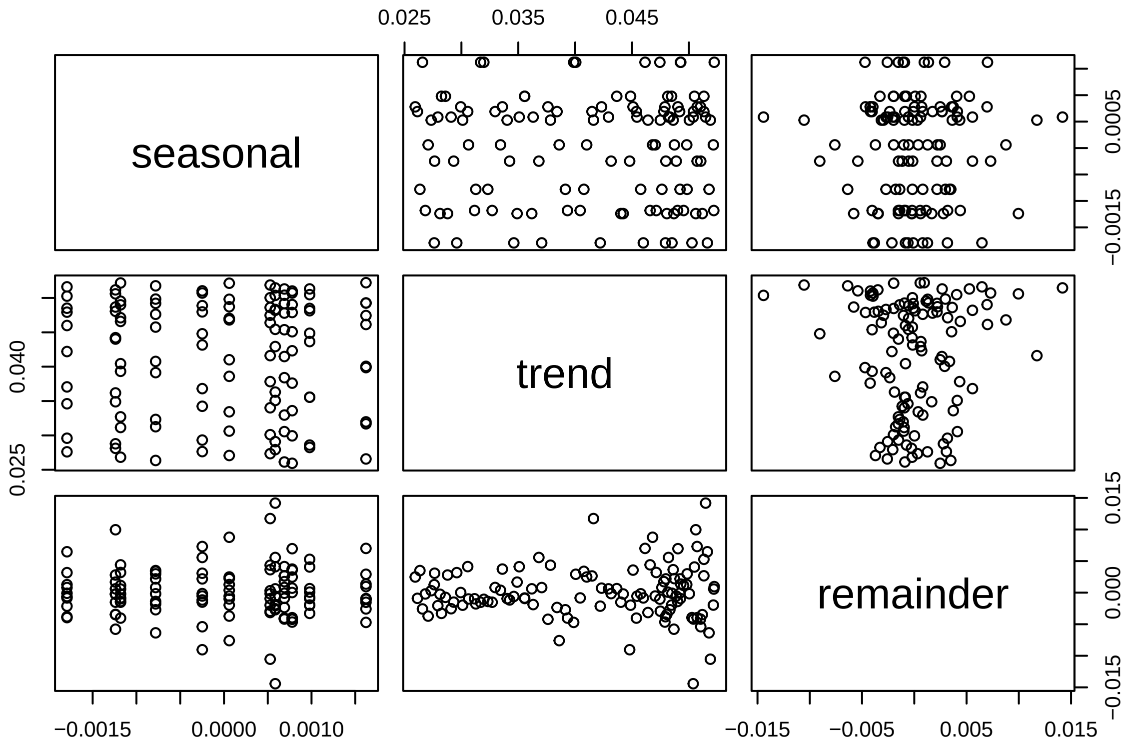



14. To accompany your graph, summarize your results in context of the research question. Include output from the statistical test in parentheses at the end of your sentence. Feel free to use multiple sentences in your interpretation.

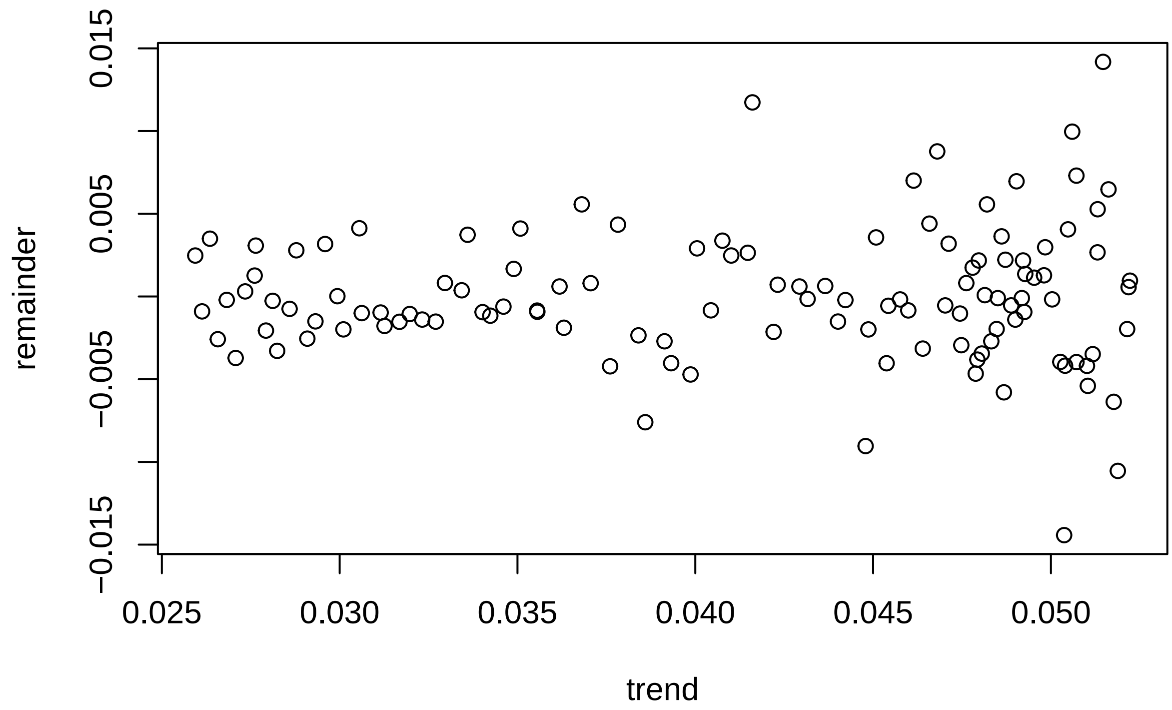
    Answer: The summer and spring months have the highest average monthly mean ozone concentrations. Throughout the study period we can see an overall trend of a slight decreasing ozone concentrations, in all months S = -7.520000e+02 (p-value = 0.16323).

15. Subtract the seasonal component from the `GaringerOzone.monthly.ts`. Hint: Look at how we extracted the series components for the EnoDischarge on the lesson Rmd file.
16. Run the Mann Kendall test on the non-seasonal Ozone monthly series. Compare the results with the ones obtained with the Seasonal Mann Kendall on the complete series.

```
#15
GaringerOzone.monthly_Components <- as.data.frame(GaringerOzone.monthly.decomp$time.series[,1:3])
plot(GaringerOzone.monthly_Components)
```




```r
GaringerOzone.monthly_noseason <- as.data.frame(GaringerOzone.monthly.decomp$time.series[,2:3])
plot(GaringerOzone.monthly_noseason)
```




```
GaringerOzone.monthly_noseason <- mutate(GaringerOzone.monthly_Components,
        Observed = GaringerOzone.monthly$meanOzone,
        Date = GaringerOzone.monthly$Date)

GaringerOzone.monthly_noseason <- GaringerOzone.monthly_noseason %>%
      mutate(NoSeasonality = (Observed - seasonal))
#16
#non-seasonal time series
GaringerOzone.monthlycomponents.ts <- ts(GaringerOzone.monthly_noseason$NoSeasonality,
                   start=c(2010, 1),
                   frequency=12)
Kendall::MannKendall(GaringerOzone.monthlycomponents.ts)
```

```
## tau = -0.101, 2-sided pvalue =0.10388
```

```
summary(GaringerOzone.monthlycomponents.ts)
```

```
##    Min. 1st Qu.  Median    Mean 3rd Qu.    Max.
## 0.02336 0.03368 0.04312 0.04149 0.04841 0.06565
```

```
#seasonal time series
GaringerOzone.monthlytrend <- Kendall:: SeasonalMannKendall(GaringerOzone.monthly.ts)
GaringerOzone.monthlytrend
```

```
## tau = -0.1, 2-sided pvalue =0.16323
```

```r
summary(GaringerOzone.monthlytrend)
```

```
## Score =  -54 , Var(Score) = 1500
## denominator =  540
## tau = -0.1, 2-sided pvalue =0.16323
```

> Answer: This p value is smaller than the p value of the non-seasonality (0.10<0.16). This makes sense as ozone levels vary with season and subtracting this component indicates weak evidence against the null hypothesis.